Tri-Lab Survey
Software Quality Practices

Software Process Assessment
Software Process Maturity Questionnaire

Based on
SEI Capability Maturity Model, Version 1.1
SEMATECH Standardized Supplier Quality Assessment
Motorola Quality System Review, Subsection 10, Software Quality Assurance

This page intentionally left blank

Questionnaire Contents

This page intentionally left blank

Section I Introduction

Questionnaire Overview

This document contains questions about the implementation of important software practices in your software organization. The questions are organized in groups of key process areas such as software project planning and software configuration management. Unless directed otherwise by the person who administers this questionnaire, please answer the questions based on your knowledge and experience in your current project, or a project list identified for reference use.

The questions address activities that should be performed within the key process area. You will be asked to indicate which activities you perform on a rating scale of 0 to 10. Guidelines for scoring your answers are included. In addition, you will be asked to provide documented evidence, as appropriate, to verify results of implemented activities.

Please read and answer all of the activity questions. If you wish to comment on any questions or qualify your answers, please use the comment spaces provided.

Project Profile Sheet

A project profile sheet should be completed for each project for which this questionnaire applies. The project profile sheets help the assessment team to better interpret your answers on the questionnaire.

Confidentiality

Your answers will be held in strict confidence by the assessment or evaluation team. Any information identifying you will be kept separate from your answers, which will be reported in combination with those of many other people. Specific answers will not be identified within your organization.

Respondent Identification

The respondent can be a knowledgeable individual or a team of knowledgeable individuals. Please indicate below who was involved in completing the questionnaire. Respondent names will be used for administrative purposes only to guide the assessment or evaluation team during response analysis and for identifying contacts during the on-site period.

Respondents’ Names:	Date:

Work Address:

Work Phone:	FAX Number:

E-mail:

Section II Software Key Process Areas Capability Assessment

Instructions

Use a pen with dark blue or black ink.

Please print or write neatly throughout the questionnaire. Please keep your ratings within the boxes.

Use the margins if you need more space for your written answers or comments, but please do not write over the rating boxes.

If a key process area is not applicable to your organization, indicate this in the comment space and do not provide any ratings.

1. Each Key Process Area has a similar format.
   - Terminology
   - Goals / Requirements Statement
   - Activity Indicators that you are to rate on a scale of 0 to 10
   - Your rating of your organization in the three evaluation dimensions: Approach, Deployment, and Results
   - Area for you to indicate evidence that will be provided representative of your activity
   - [Optional] Separate sheet of comments on other approaches / activities you are using to meet the requirement, or clarify your plans, or discuss the application of this requirement to your environment

2. Terminology

Each Key Process Area has a small section of terms that are frequently used. This should assist the respondent in understanding the general requirement and activity indicators for the Area. A glossary of general software process improvement terms is in Appendix C.

3. Goals / Requirements Statement

Each Key Process Area has a brief summary statement that reflects the overall requirement or goals of the Area. In order for the organization to be successful in meeting the requirement / goals, the indicated activities should be achieved.

4. Activity Indicators Ratings

Each Key Process Area includes activities that reflect what the organization should be conducting to achieve success in this Area. Each activity has a rating box that the respondent should complete with an integer score from 0 to 10. General guidelines for rating the key activities is in Appendix A.

5. Evaluation Dimension Ratings

Each Key Process Area has three evaluation dimensions that should be rated by the respondent on a scale of 0 to 10. Guidelines for the ratings are in Appendix A. These evaluation dimensions include:

- Approach: criteria here are the organization’s commitment to and management’s support for the practices as well as the organization’s ability to implement the practices of the Key Process Area.
- Deployment: the breadth and consistency of practice implementation across project areas are the key criteria.
- Results: criteria are the breadth and consistency of positive results over time and across project areas.

1. Evidence

An area is provided to indicate any evidence that will be provided for this Key Process Area. Guidelines for selection of evidence to support each Key Process Areas is provided in Appendix B.

2. [Optional] Comments on other organizational approaches

If there are other approaches, organization plans, or information on application of activities that provides insight into how your organization meets or intends to satisfy this Key Process Area’s requirements, please provide your comments in the space provided or on a separate sheet.

This page intentionally left blank

KPA.01 Requirements Management

Terminology

The purpose of **Requirements Management** is to establish a common understanding between the customer and the software project of the customer's requirements that will be addressed by the software project. Requirements Management involves establishing and maintaining an agreement with the customer on the requirements for the software project. The agreement covers both the technical and non technical (e.g., delivery dates) requirements. The agreement forms the basis for estimating, planning, performing, and tracking the software project's activities throughout the software life cycle. Whenever the system requirements allocated to software are changed, the affected software plans, work products, and activities are adjusted to remain consistent with the updated requirements.

allocated requirements (system requirements allocated to software) - The subset of the system requirements that are to be implemented in the software components of the system. The allocated requirements are a primary input to the software development plan. Software requirements analysis elaborates and refines the allocated requirements and results in software requirements which are documented.

software plans - The collection of plans, both formal and informal, used to express how software development and/or maintenance activities will be performed. Examples of plans that could be included: software development plan, software quality assurance plan, software configuration management plan, software test plan, risk management plan, and process improvement plan.

software engineering group - The collection of individuals (both managers and technical staff) who have responsibility for software development and maintenance activities (i.e., requirements analysis, design, code, and test) for a project. Groups performing software-related work, such as the software quality assurance group, the software configuration management group, and the software engineering process group, are not included in the software engineering group.

software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

Goals / Requirements

System requirements allocated to software are controlled to establish a baseline for software engineering and management use. Software plans, products, and activities are kept consistent with the system requirements allocated to software.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. The software engineering group reviews the allocated requirements before they are incorporated into the software project.
	2. The software engineering group used the allocated requirements as the basis for software plans, work products, and activities.
	3. Changes to the allocated requirements are reviewed and incorporated into the software project.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Requirements Management		
Deployment of Requirements Management		
Results Achieved in Requirements Management		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

KPA.02 Software Project Planning

Terminology

The purpose of **Software Project Planning** is to establish reasonable plans for performing the software engineering activities and for managing the software project. Software Project Planning involves developing estimates for the work to be performed, establishing the necessary commitments, and defining the plan to perform the work.

commitment - A pact that is freely assumed, visible, and expected to be kept by all parties.
documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]
event-driven review/activity - A review or activity that is performed based on the occurrence of an event within the project (e.g., a formal review or the completion of a life-cycle stage).
periodic review/activity - A review /activity that occurs at a specified regular time interval, rather than at the completion of major events.
software engineering group - The collection of individuals (both managers and technical staff) who have responsibility for software development and maintenance activities (i.e., requirements analysis, design, code, and test) for a project. Groups performing software-related work, such as the software quality assurance group, the software configuration management group, and the software engineering process group, are not included in the software engineering group.
software plans - The collection of plans, both formal and informal, used to express how software development and/or maintenance activities will be performed. Examples of plans that could be included: software development plan, software quality assurance plan, software configuration management plan, software test plan, risk management plan, and process improvement plan
software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user. (See *software product* for contrast.)
.

Goals / Requirements

Software estimates are documented for use in planning and tracking the software project. Software project activities and commitments are planned and documented. Affected groups and individuals agree to their commitments related to the software project.

Activity Indicators (rate your organization's capability on an integer scale from 0 to 10)

	1. The software engineering group participates on the project proposal team.
	2. Software project planning is initiated in the early stages of, and in parallel with, the overall project planning.
	3. The software engineering group participates with other affected groups in the overall project planning throughout the project's life.
	4. Software project commitments made to individuals and groups external to the organization are reviewed with senior management according to a documented procedure.
	5. A software life cycle with predefined stages of manageable size is identified or defined.
	6. The project's software development plan is developed according to a documented procedure.
	7. The plan for the software project is documented.
	8. Software work products that are needed to establish and maintain control of the software project are identified.
	9. Estimates for the size of the software work products (or changes to the size of software work products) are derived according to a documented procedure.
	10. Estimates for the software project's effort and costs are derived according to a documented procedure.
	11. Estimates for the project's critical computer resources are derived according to a documented procedure.
	12. The project's software schedule is derived according to a documented procedure.

	13. The software risks associated with the cost, resource, schedule, and technical aspects of the project are identified, assessed, and documented.
	14. Plans for the project's software engineering facilities and support tools are prepared.
	15. Software planning data are recorded.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Software Project Planning		
Deployment of Software Project Planning		
Results Achieved in Software Project Planning		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

Project plan lists risk mitigation alternatives

KPA.03 Software Project Tracking and Oversight

Terminology

The purpose of **Software Project Tracking and Oversight** is to provide adequate visibility into actual progress sop that management can take corrective actions when the software project's performance deviates significantly from the software plans. Corrective actions may include revising the software development plan to reflect the actual accomplishments and replanning the remaining work or taking actions to improve the performance. Software Project Tracking and Oversight involves tracking and reviewing the software accomplishments and results against documented estimates, commitments, and plans, and adjusting these plans based on the actual accomplishments and results.

commitment - A pact that is freely assumed, visible, and expected to be kept by all parties.
differ significantly - A difference that is sufficient to warrant corrective action.
documented procedure - A written description of a course of action to be taken to perform a given. [IEEE-STD-610]
formal review - A formal meeting at which a product is presented to the end user, customer, or other interested parties for comment and approval. It can also be a review of the management and technical activities and of the progress of the project.
software engineering group - The collection of individuals (both managers and technical staff) who have responsibility for software development and maintenance activities (i.e., requirements analysis, design, code, and test) for a project. Groups performing software-related work, such as the software quality assurance group, the software configuration management group, and the software engineering process group, are not included in the software engineering group.
software plans - The collection of plans, both formal and informal, used to express how software development and/or maintenance activities will be performed. Examples of plans that could be included: software development plan, software quality assurance plan, software configuration management plan, software test plan, risk management plan, and process improvement plan.
software-related group - A collection of individuals (both managers and technical staff) representing a software engineering discipline that supports, but is not directly responsible for, software development and/or maintenance. Examples of software engineering disciplines include software quality assurance and software configuration management.
software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

Goals / Requirements

Actual results and performances are tracked against the software plans. Corrective actions are taken and managed to closure when actual results and performance deviate significantly from the software plans. Changes to software commitments are agreed to by the affected groups and individuals.

Activity Indicators (rate your organization's capability on an integer scale from 0 to 10)

	1 1. A documented software development plan is used for tracking the software activities and communicating status.
	2 2. The project's software development plan is revised according to a documented procedure.
	3 3. Software project commitments and changes to commitments made to individuals and groups external to the organization are reviewed with senior management according to a documented procedure.
	4 4. Approved changes to commitments that affect the software project are communicated to the members of the software engineering group and other software-related groups.
	5 5. The size of the software work products (or size of the changes to the software work products) are tracked, and corrective actions are taken as necessary.
	6 6. The project's software effort and costs are tracked, and corrective actions are taken as necessary.
	7 7. The project's critical computer resources are tracked, and corrective actions are taken as necessary.
	8 8. The project's software schedule is tracked, and corrective actions are taken as necessary.
	9 9. Software engineering technical activities are tracked, and corrective actions are taken as necessary.
	10 10. The software risks associated with cost, resource, schedule, and technical aspects of the project are tracked.

	11 11. Actual measurement data and replanning data for the software project are recorded.
	12 12. The software engineering group conducts periodic internal reviews to track technical progress, plans, performance, and issues against the software development plan.
	13 13. Formal reviews to address the accomplishments and results of the software project are conducted at selected project milestones according to a documented procedure.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Software Project Tracking and Oversight		
Deployment of Software Project Tracking and Oversight		
Results Achieved in Software Project Tracking and Oversight		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization’s environment. Use a separate sheet if preferred.

KPA.04 Software Subcontract Management

Terminology

The purpose of **Software Subcontract Management** is to select qualified software subcontractors and manage them effectively. Software Subcontract management involves selecting a software subcontractor, establishing commitments with the subcontractor, and tracking and reviewing the subcontractor's performance and results. These practices cover the management of a software (only) subcontract, as well as the management of the software component of a subcontract that includes software, hardware, and possibly other system components.

documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]
event-driven review/activity - A review or activity that is performed based on the occurrence of an event within the project (e.g., a formal review or the completion of a life-cycle stage).
prime contractor - An individual, partnership, corporation, or association that administers a subcontract to design, develop, and/or manufacture one or more products.
software development plan - The collection of plans that describe the activities to be performed for the software project. It governs the management of the activities performed by the software engineering group for a software project. It is not limited to the scope of any particular planning standard, such as DOD-STD-2167A and IEEE-STD-1058, which may use similar terminology.
software quality assurance - (1) A planned and systematic pattern of all actions necessary to provide adequate confidence that a software work product conforms to established technical requirements. (2) A set of activities designed to evaluate the process by which software work products are developed and/or maintained. [Derived from IEEE-STD-610]
subcontract manager - A manager in the prime contractor's organization who has direct responsibility for administering and managing one or more subcontracts.
subcontractor - An individual, partnership, corporation, or association that contracts with an organization (i.e., the prime contractor) to design, develop, and/or manufacture one or more products.

Goals / Requirements

The prime contractor selects qualified software subcontractors. The prime contractor and the software subcontractor agree to their commitments to each other. The prime contractor and the software subcontractor maintain ongoing communications. The prime contractor tracks the software subcontractor’s actual results and performance against its commitments.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. The work to be subcontracted is defined and planned according to a documented procedure.
	2. The software subcontractor is selected, based on an evaluation of the subcontract bidders’ ability to perform the work, according to a documented procedure.
	3. The contractual agreement between the prime contractor and the software subcontractor is used as the basis for managing the subcontract.
	4. A documented subcontractor’s software development plan is reviewed and approved by the prime contractor.
	5. A documented and approved subcontractor’s software development plan is used for tracking the software activities and communicating status.
	6. Changes to the software subcontractor’s statement of work, subcontract terms and conditions, and other commitments are resolved according to a documented procedure.
	7. The prime contractor’s management conducts periodic status / coordination reviews with the software subcontractor’s management.
	8. Periodic technical reviews and interchanges are held with the software subcontractor.
	9. Formal reviews to address the subcontractor’s software engineering accomplishments and results are conducted at selected milestones according to a documented procedure.
	10. The prime contractor’s software quality assurance group monitors the subcontractor’s software quality assurance activities according to a documented procedures.

	11. The prime contractor's software configuration management group monitors the subcontractor's activities for software configuration management according to a documented procedure.
	12. The prime contractor conducts acceptance testing as part of the delivery of the subcontractor's software products according to a documented procedure
	13. The software subcontractor's performance is evaluated on a periodic basis, and the evaluation is reviewed with the subcontractor.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Software Subcontract Management		
Deployment of Software Subcontract Management		
Results Achieved in Software Subcontract Management		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

We have NO control over our subcontractors at all!

KPA.05 Software Quality Assurance

Terminology

The purpose of the **Software Quality Assurance** (SQA) is to provide management with appropriate visibility into the process being used by the software project and of the products being built. Software Quality Assurance involves reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards and providing the software project and other appropriate managers with the results of these reviews and audits.

audit - An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria. [IEEE-STD-610]

periodic review/activity - A review/activity that occurs at a specified regular time interval, rather than at the completion of major events.

procedure - A written description of a course of action to be taken to perform a given task. [IEEE-STD-610]

quality - (1) The degree to which a system, component, or process meets specified requirements. (2) The degree to which a system, component, or process meets customer or user needs or expectations. [IEEE-STD-610]

software engineering group - The collection of individuals (both managers and technical staff) who have responsibility for software development and maintenance activities (i.e., requirements analysis, design, code, and test) for a project. Groups performing software-related work, such as the software quality assurance group, the software configuration management group, and the software engineering process group, are not included in the software engineering group.

software quality assurance (SQA) - (1) A planned and systematic pattern of all actions necessary to provide adequate confidence that a software work product conforms to established technical requirements. (2) A set of activities designed to evaluate the process by which software work products are developed and/or maintained.

software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

standard - Mandatory requirements employed and enforced to prescribe a disciplined, uniform approach to software development.

Goals / Requirements

Software quality assurance activities are planned. Adherence of software products and activities to the applicable standards, procedures, and requirements is verified objectively. Affected groups and individuals are informed of software quality assurance activities and results. Noncompliance issues that cannot be resolved within the software project are addressed by senior management.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. An SQA plan is prepared for the software project according to a documented procedure.
	2. The SQA group’s activities are performed in accordance with the SQA plan.
	3. The SQA group participates in the preparation and review of the project’s software development plan, standards, and procedures.
	4. The SQA group reviews the software engineering activities to verify compliance.
	5. The SQA group audits designated software work products to verify compliance.
	6. The SQA group periodically reports the results of its activities to the software engineering group.
	7. Deviations identified in the software activities and software work products are documented and handled according to a documented procedure.
	8. The SQA group conducts periodic reviews of its activities and findings with the customer’s SQA personnel, as appropriate.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Software Quality Assurance		
Deployment of Software Quality Assurance		
Results Achieved in Software Quality Assurance		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization’s environment. Use a separate sheet if preferred.

KPA.06 Software Configuration Management

Terminology

The purpose of **Software Configuration Management** (SCM) is to establish and maintain the integrity of the products of the software project throughout the project's software life cycle. Software Configuration Management involves identifying the configuration of the software (e.g., selected software work products and their descriptions) at given points in time, systematically controlling changes to the configuration, and maintaining the integrity and traceability of the configuration throughout the software life cycle. The work products placed under software configuration management include the software products that are delivered to the customer and the items that are identified with or required to create these software products.

audit - An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria. [IEEE-STD-610]

configuration item - An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process. {IEEE-STD-610]

configuration management - A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements. [IEEE-STD-610]

documented procedure - A written description of a course of action to be taken to perform a given task. [IEEE-STD-610]

software baseline - A set of configuration items (software documents and software components) that has been formally reviewed and agreed upon, that thereafter serves as the basis for future development, and that can be changed only through formal change control procedures

software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

Goals / Requirements

Software configuration management activities are planned. Selected software work products are identified, controlled, and available. Changes to identified software work products are controlled. Affected groups and individuals are informed of the status and content of software baselines.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. A SCM plan is prepared for each software project according to a documented procedure.
	2. A documented and approved SCM plan is used as the basis for performing the SCM activities.
	3. A configuration management library system is established as a repository for the software baselines.
	4. The software work products to be placed under configuration management are identified.
	5. Change requests and problem reports for all configuration items / units are initiated, recorded, reviewed, approved, and tracked according to a documented procedure.
	6. Changes to baselines are controlled according to a documented procedure.
	7. Products from the software baseline library are created and their release is controlled according to a documented procedure.
	8. The status of configuration items / units is recorded according to a documented procedure.
	9. Standard reports documenting the SCM activities and the contents of the software baseline are developed and made available to affected groups and individuals.
	10. Software Baseline audits are conducted according to a documented procedure.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Software Configuration Management		
Deployment of Software Configuration Management		
Results Achieved in Software Configuration Management		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization’s environment. Use a separate sheet if preferred.

KPA.07 Organization Process Focus

Terminology

The purpose of **Organization Process Focus** is to establish the organizational responsibility for software process activities that improve the organization's overall software process capability. Organization Process Focus involves developing and maintaining an understanding of the organization's and projects' software processes and coordinating the activities to assess, develop, maintain, and improve these processes. The organization provides long-term commitments and resources to coordinate the development and maintenance of the software processes across current and future software projects via a group such as a software engineering process group. This group is responsible for the organization's software process activities.

periodic review/activity - A review/activity that occurs at a specified regular time interval, rather than at the completion of major events.

software process - A set of activities, methods, practices, and transformations that people use to develop and maintain software and the associated products (e.g., project plans, design documents, code, test cases, and user manuals)

software process assessment - An appraisal by a trained team of software professionals to determine the state of an organization's current software process, to determine the high-priority software process-related issues facing an organization, and to obtain the organizational support for software process improvement.

training program - The set of related elements that focus on addressing an organization's training needs. It includes an organization's training plan, training materials, development of training, conduct of training, training facilities, evaluation of training, and maintenance of training records.

Goals / Requirements

Software process development and improvement activities are coordinated across the organization. The strengths and weaknesses of the software processes used are identified relative to a process standard. Organization-level process development and improvement activities are planned.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. The software process is assessed periodically, and action plans are developed to address the assessment findings.
	2. The organization develops and maintains a plan for its software process development and improvement activities.
	3. The organization’s and projects’ activities for developing and improving their software processes are coordinated at the organization level.
	4. The use of the organization’s software process database is coordinated at the organizational level.
	5. New processes, methods, and tools in limited use in the organization are monitored, evaluated, and, where appropriate, transferred to other parts of the organization.
	6. Training for the organization’s and projects’ software processes is coordinated across the organization.
	7. The groups involved in implementing the software processes are informed of the organization’s and projects’ activities for software process development and improvement.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Organization Process Focus		
Deployment of Organization Process Focus		
Results Achieved in Organization Process Focus		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

KPA.08 Organization Process Definition

Terminology

The purpose of **Organization Process Definition** is to develop and maintain a usable set of software process assets that improve process performance across the projects and provide a basis for cumulative, long-term benefits to the organization. Organization Process Definition involves developing and maintaining the organization's standard software process, along with related process assets, such as descriptions of software life cycles, process tailoring guidelines and criteria, the organization's software process database, and a library of software process related documentation.

audit - An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria.

documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]

organization's standard software process - The operational definition of the basic process that guides the establishment of a common software process across the software projects in an organization. It describes the fundamental software process elements that each software project is expected to incorporate into its defined software process. It also describes the relationships (e.g., ordering and interfaces) between these software process elements.

software life cycle - The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and, sometimes, retirement phase. [IEEE-STD-610]

Goals / Requirements

A standard software process for the organization is developed and maintained.
Information related to the use of the organization’s standard software process by the software projects is collected, reviewed, and made available.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. The organization’s standard software process is developed and maintained according to a documented procedure.
	2. The organization’s standard software process is documented according to established organization standards.
	3. Descriptions of software life cycles that are approved for use by projects are documented and maintained.
	4. Guidelines and criteria for the projects’ tailoring of the organization’s standard software process are developed and maintained.
	5. The organization’s software process database is established and maintained.
	6. A library of software process-related documentation is established and maintained.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Organization Process Definition		
Deployment of Organization Process Definition		
Results Achieved in Organization Process Definition		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

KPA.09 Training Program

Terminology

The purpose of the **Training Program** key process area is to develop the skills and knowledge of individuals so they can perform their roles effectively and efficiently. Training Program involves first identifying the training needed by the organization, projects, and individuals, then developing or procuring training to address the identified needs. Some skills are effectively and efficiently imparted through informal vehicles (e.g., on-the-job training and informal mentoring), whereas other skills need more formal training vehicles (e.g., classroom training and guided self-study) to be effectively and efficiently imparted. The appropriate vehicles are selected and used.

documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]

software project - An undertaking requiring concerted effort, which is focused on analyzing, specifying, designing, developing, testing, and/or maintaining the software components and associated documentation of a system. A software project may be part of a project building a hardware/software system.

standard - Mandatory requirements employed and enforced to prescribe a disciplined uniform approach to software development. [Derived from IEEE-STD-610]

training program - The set of related elements that focus on addressing an organization's training needs. It includes an organization's training plan, training materials, development of training, conduct of training, training facilities, evaluation of training, and maintenance of training records.

training waiver - A written approval exempting an individual from training that has been designated as required for a specific role. The exemption is granted because it has been objectively determined that the individual already possesses the needed skills to perform the role.

Goals / Requirements

Training activities are planned. Training for developing the skills and knowledge needed to perform software management and technical roles is provided. Individuals in the software engineering group and software-related groups receive the training necessary to perform their roles.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. Each software project develops and maintains a training plan that specifies its training needs.
	2. The organization’s training plan is developed and revised according to a documented procedure.
	3. The training for the organization is performed in accordance with the organization’s training plan.
	4. Training courses prepared at the organization level are developed and maintained according to organization standards.
	5. A waiver procedure for required training is established and used to determine whether individuals already possess the knowledge and skills required to perform in their designated roles.
	6. Records of training are maintained.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Training Program		
Deployment of Training Program		
Results Achieved in Training Program		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

KPA.10 Integrated Software Management

Terminology

The purpose of **Integrated Software Management** is to integrate the software engineering and management activities into a coherent, defined software process that is tailored from the organization's standard software process and related process assets. Integrated Software Management involves developing the project's defined software process and managing the software project using this defined software process. The project's defined software process is tailored from the organization's standard software process to address the specific characteristics of the project. The software development plan is based on the project's defined software process and describes how the activities of the project's defined software process will be implemented and managed.

customer - The individual or organization that is responsible for accepting the product and authorizing payment to the developing organization.

documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]

end user - The individual or group who will use the system for its intended operational use when it is deployed in its environment.

organization's standard software process - The operational definition of the basic process that guides the establishment of a common software process across the software projects in an organization. It describes the fundamental software process elements that each software project is expected to incorporate into its defined software process. It also describes the relationships (e.g., ordering and interfaces) between these software process elements.

periodic review/activity - A review /activity that occurs at a specified regular time interval, rather than at the completion of major events.

project's defined software process - The operational definition of the software process used by a project. The project's defined software process is a well-characterized and understood software process, described in terms of software standards, procedures, tools, and methods. It is developed by tailoring the organization's standard software process to fit the specific characteristics of the project.

software development plan - The collection of plans that describe the activities to be performed for the software project. It governs the management of the activities performed by the software engineering group for a software project. It is not limited to the scope of any particular planning standard, such as DOD-STD-2167A and IEEE-STD-1058, which may use similar terminology.

software project - An undertaking requiring concerted effort, which is focused on analyzing, specifying, designing, developing, testing, and/or maintaining the software components and associated documentation of a system. A software project may be part of a project building a hardware/software system.

software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user. (See *software product* for contrast.)

tailoring - To modify a process, standard, or procedure to better match process or product requirements.

Goals / Requirements

The project's defined software process is a tailored version of the organization's standard software process. The project is planned and managed according to the project's defined software process.

Activity Indicators (rate your organization's capability on an integer scale from 0 to 10)

	1. The project's defined software process is developed by tailoring the organization's standard software process according to a documented procedure.
	2. Each project's defined software process is revised according to a documented procedure.
	3. The project's software development plan, which describes the use of the project's defined software process, is developed and revised according to a documented procedure.
	4. The software project is managed in accordance with the project's defined software process.
	5. The organization's software process database is used for software planning and estimating.
	6. The size of the software work products (or size of changes to the software work products) is managed according to a documented procedure.

	7. The project’s software effort and costs are managed according to a documented procedure.
	8. The project’s critical computer resources are managed according to a documented procedure.
	9. The critical dependencies and critical paths of the project’s software schedule are managed according to a documented procedure.
	10. The Project’s software risks are identified, assessed, documented, and managed according to a documented procedure.
	11. Reviews of the software project are periodically performed to determine the actions needed to bring the software project’s performance and results in line with the current and projected needs of the business, customer, and end users, as appropriate.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Integrated Software Management		
Deployment of Integrated Software Management		
Results Achieved in Integrated Software Management		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization’s environment. Use a separate sheet if preferred.

KPA.11 Software Product Engineering

Terminology

The purpose of the **Software Product Engineering** is to consistently perform a well-defined engineering process that integrates all the software engineering activities to produce correct, consistent software products effectively and efficiently. Software Product Engineering involves performing the engineering tasks to build and maintain the software using the project's defined software process and appropriate methods and tools. The software engineering tasks include analyzing the system requirements allocated to software, developing the software architecture's designing the software, implementing the software in the code, and testing the software to verify that it satisfies the specified requirements.

peer review - A review of a software work product, following defined procedures, by peers of the producers of the product for the purpose of identifying defects and improvements.
project's defined software process - The operational definition of the software process used by a project. The project's defined software process is a well-characterized and understood software process, described in terms of software standards, procedures, tools, and methods. It is developed by tailoring the organization's standard software process to fit the specific characteristics of the project.
software plans - The collection of plans, both formal and informal, used to express how software development and/or maintenance activities will be performed. Examples of plans that could be included: software development plan, software quality assurance plan, software configuration management plan, software test plan, risk management plan, and process improvement plan.
software product - The complete set, or any of the individual items of the set, of computer programs, procedures, and associated documentation and data designated for delivery to a customer or end user. [IEEE-STD-610]
software requirement - A condition or capability that must be met by software needed by a user to solve a problem or achieve an objective. [IEEE-STD-610]
software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

Goals / Requirements
The software engineering tasks are defined, integrated, and consistently performed to produce the software.
Software work products are kept consistent with each other.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. Appropriate software engineering methods and tools are integrated into the project’s defined software process.
	2. The software requirements are developed, maintained, documented. and verified by systematically analyzing the allocated requirements according to the project’s defined software process.
	3. The software design is developed, maintained, documented, and verified, according to the project’s defined software process, to accommodate the software requirements and to form the framework for coding.
	4. The software code is developed, maintained, documented, and verified, according to the project’s defined software process, to implement the software requirements and software design.
	5. Software testing is performed according to the project’s defined software process.
	6. System and acceptance testing of the software are planned and performed to demonstrate that the software satisfies its requirements.
	7. The documentation that will be used to operate and maintain the software is developed and maintained according to the project’s defined software process.
	8. Data on defects identified in peer reviews and testing are collected and analyzed according to the project’s defined software process.
	9. Consistency is maintained across software work products, including the software plans, process descriptions, allocated requirements, software requirements, software design, code, test plans, and test procedures.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Software Product Engineering		
Deployment of Software Product Engineering		
Results Achieved in Software Product Engineering		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization’s environment. Use a separate sheet if preferred.

KPA.12 Intergroup Coordination

Terminology

The purpose of **Intergroup Coordination** is to establish a means for the software engineering group to participate actively with the other engineering groups so the project is better able to satisfy the customer's needs effectively and efficiently. Intergroup Coordination involves the software engineering group's participation with other project engineering groups to address system-level requirements, objectives, and issues. Representatives of the project's engineering groups participate in establishing the system-level requirements, objectives, and plans by working with the customer and end users, as appropriate. these requirements, objectives, and plans become the basis for all engineering activities.

customer - The individual or organization that is responsible for accepting the product and authorizing payment to the developing organization.
commitment - A pact that is freely assumed, visible, and expected to be kept by all parties.
documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]
end user - The individual or group who will use the system for its intended operational use when it is deployed in its environment.
engineering group - A collection of individuals (both managers and technical staff) representing an engineering discipline. Examples of engineering disciplines include systems engineering, hardware engineering, system test, software engineering, software configuration management, and software quality assurance.
software engineering group - The collection of individuals (both managers and technical staff) who have responsibility for software development and maintenance activities (i.e., requirements analysis, design, code, and test) for a project. Groups performing software-related work, such as the software quality assurance group, the software configuration management group, and the software engineering process group, are not included in the software engineering group.

Goals / Requirements

The customer’s requirements are agreed to by all affected groups. The commitments between the engineering groups are agreed to by the affected groups. The engineering groups identify, track, and resolve intergroup issues.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. The software engineering group and the other engineering groups participate with the customer and end users, as appropriate, to establish the system requirements.
	2. Representatives of the project’s software engineering group work with representatives of the other engineering groups to monitor and coordinate technical activities and resolve technical issues.
	3. A documented plan is used to communicate intergroup commitments and to coordinate and track the work performed.
	4. Critical dependencies between engineering groups are identified, negotiated, and tracked according to a documented procedure.
	5. Work products produced as input to other engineering groups are reviewed by representatives of the receiving groups to ensure that the work products meet their needs.
	6. Intergroup issues not resolvable by the individual representatives of the project engineering groups are handled according to a documented procedure.
	7. Representatives of the project engineering groups conduct periodic technical reviews and interchanges.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Intergroup Coordination		
Deployment of Intergroup Coordination		
Results Achieved in Intergroup Coordination		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

KPA.13 Peer Reviews

Terminology

The purpose of **Peer Reviews** is to remove defects from the software work products early and efficiently. An important corollary effect is to develop a better understanding of the software work products and of defects that might be prevented. Peer Reviews involve a methodical examination of software work products by the producers' peers to identify defects and areas where changes are needed. The specific products that will undergo a peer review are identified in the project's defined software process and scheduled as part of the software project planning activities.

audit - An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria.
defect - A flaw in a system or system component that causes the system or component to fail to perform its required function. A defect, if encountered during execution, may cause a failure of the system.
documented procedure - A written description of a course of action to be taken to perform a given task [IEEE-STD-610]
peer review - A review of a software work product, following defined procedures, by peers of the producers of the product for the purpose of identifying defects and improvements.
software quality assurance (SQA) - (1) A planned and systematic pattern of all actions necessary to provide adequate confidence that a software product conforms to established technical requirements. (2) A set of activities designed to evaluate the process by which software work products are developed and/or maintained.
software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user.

Goals / Requirements

Peer review activities are planned. Defects in the software work products are identified and removed.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. Peer reviews are planned, and the plans are documented.
	2. Peer reviews are performed according to a documented procedure.
	3. Data on the conduct and results of the peer reviews are recorded.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Peer Reviews		
Deployment of Peer Reviews		
Results Achieved in Peer Reviews		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization’s environment. Use a separate sheet if preferred.

ASCI KPA Validation and Verification

Terminology

The purpose of **Validation and Verification** is to plan, execute, and document activities that establish confidence in the resultant ASCI product. The integration of formalized V&V activities with other software processes is essential to assure confidence in the resultant ASCI product. As each code product has different origins in data and functionality, the relevance of data and data sources, legacy software, similar legacy codes, and availability of qualified peer review groups will be different for each code product. Thus the methods used to assert Validation will be different for each code product. Thus it is important for both near term and long term that a V&V Implementation plan be formulated and executed following the "Strategic Computing & Simulation Validation &Verification Program Plan".

Validation - The process of determining the degree to which a computer simulation is an accurate representation of the real world.
Verification - the process of determining that a computer simulation correctly represents the conceptual model and its solution.

V&V Implementation Plans - The collection of plans, both formal and informal, used to express how Validation and Verification activities will be performed. Examples of plans that could be included: software validation and verification plan, software development plan, software quality assurance plan, software configuration management plan, software test plan, risk management plan, and process improvement plan.

Data validation - Necessary steps to assure data, against which code can be code output can be compared. Steps may include assurance that all relevant data is available, identification of critical gaps in the data, estimates of uncertainty in the data and limits of applicability, and documentation of the validation steps, for example.

Goals / Requirements

Goals and Requirements of the V&V activity is to establish confidence in the simulations supporting the Stockpile Stewardship Program through systematic demonstration and documentation of predictive capability of the codes and their underlying models.

Activity Indicators (rate your organization’s capability on an integer scale from 0 to 10)

	1. V&V Implementation Plan is known and followed by program participants.
	2. Data validation activities are documented according to the V&V Implementation Plan.
	3. A documented test plan is followed at multiple phases of the V&V effort.
	4. Validation Requirements are documented.
	5. Testing is performed in accordance with documented test plans, test cases and detailed test procedures.
	6. Verification and Validation is included as a software process element that contains a set of closely related tasks that are well defined and bounded.
	7. Verification and Validation costs are specifically identified as such in software product cost estimates.
	8. Peer reviews include Verification and Validation work products.

Rate your approach, deployment, and results for this process area on an integer scale from 0 to 10, and identify evidence below that will be provided to support your ratings for this process area.

	Rating (0-10)	**Evidence**
Approach to Validation and Verification		
Deployment of Validation and Verification		
Results Achieved in Validation and Verification		

[Optional] Provide comments on other approaches, plans, or applicability of this process area to your organization's environment. Use a separate sheet if preferred.

Appendix A
Guidelines to Rate KPA Activities[1]

Score	Key Activity Evaluation Dimensions		
	Approach	Deployment	Results
Poor (0)	No Management recognition of need No organizational ability No organizational commitment Practice not evident	No part of the organization uses the practice No part of the organization shows interest	Ineffective
Weak (2)	Management has begun to recognize the need Support items for the practice start to be created A few parts of the organization are able to implement the practice	Fragmented use Inconsistent use Deployed in some parts of the organization Limited monitoring/verification of use	Spotty results Inconsistent results Some evidence of effectiveness for some parts of the organization

1. **M. Daskalantonakis, Motorola, " Achieving Higher SEI Levels," IEEE Computer, Vol. 27, No. 7, pages 17-24, July 1994.**

Fair (4)	Wide but not complete commitment by management Road map for practice implementation defined Several supporting items for the practice in place	Less fragmented use Some consistency in use Deployed in some major parts of the organization Monitoring/verification of use for several parts of the organization	Consistent and positive results for several parts of the organization Inconsistent results for other parts of the organization
Marginally Qualified (6)	Some management commitment; some management becomes proactive Practice implementation well under way across parts of the organization Supporting items in place	Deployed in some parts of the organization Mostly consistent use across many parts of the organization Monitoring/verification for almost all parts of the organization	Positive measurable results in most parts of the organization Consistently positive results over time across many parts of the organization
Qualified (8)	Total management commitment Majority of management is proactive Practice established as an integral part of the process Supporting items encourage and facilitate the use of the practice	Deployed in almost all parts of the organization Mostly consistent use across almost all parts of the organization Monitoring/verification for almost all parts of the organization	Positive measurable results in almost all parts of the organization Consistently positive results over time across almost all parts of the organization
Outstanding (10)	Management provides zealous leadership and commitment Organizational excellence in the practice recognized even outside the company	Pervasive and consistent deployment across al parts of the organization Consistent use across all parts of the organization Monitoring/verification for all parts of the organization	Requirements exceeded Consistently world-class Counsel sought by others

This page intentionally left blank

Appendix B
Guidelines for Selecting KPA Supporting Evidence

The following are suggestions for possible evidence to support the organization's capability within each Key Process Area (KPA) . This evidence includes existing organizational documentation (intended to be applied as guidance or standards for all projects) as well as specific project documentation applicable to one or more of the projects selected to be part of the assessment process. This evidence is not meant to be a comprehensive list, but should be helpful.

Level 2 Repeatable

KPA.01 Requirements Management

1. Organizational standard software process definition (requirements analysis section and traceability matrix)
    Organizational documentation on Software Development
    Organizational documentation on Software Guidelines
    Organizational documentation on Preferred Process for Software Development
    Subprocess definitions that are standard for the organization, such as Software Inspections
    Organization standards such as documentation templates, security/safety plans, software quality / configuration management plans, coding style and language features, and so forth.
2. Project software process definition (requirements analysis section and traceability matrix)
3. Project system to software requirements traceability matrix
    Appendix in software requirements specification
    Data base / flow format as part of automated tool set
4. Software requirements specification documentation
5. Software requirements to test case trace matrix
6. Automated support tools for requirements traceability / tracking.

KPA.02 Software Project Planning

7. Organizational standard software process definition (planning methods)
    Organizational policy statements on Software Management
    Organization Software Management Plan
    Organization Business Plans (past year, current year)
    Organization Conduct of Operations Documentation (past year, current year)
8. Project software process definition (planning methods)
9. Project software development plan
10. Project task schedule charts, project resource utilization (labor, direct costs, etc.) charts
    Estimates and actuals
11. Project documentation from internal reviews
12. Project documentation from formal reviews with the customer
13. Automated support tools for software project management / planning.

KPA.03 Software Project Tracking and Oversight

14. Software project planning evidence
15. Charts, tables, any project documentation that illustrates over time how actuals (task completion, major milestone completion, cost in $, cost in labor) compared with the estimated.
16. Documentation that describes how project changes are/were tracked, and how project impact because of the changes is determined (e.g., project risk, etc.)
    Organization Software Management Plan (risk section)
    Software risk evaluations (internal and external such as conducted by Software Engineering Institute)
17. Software change request tracking charts (number open, closed)
18. Automated support tools for project cost estimation, actual vs estimated graphical display, pert charts.

KPA.04 Software Subcontract Management

19. Organizational standard software process definition (planning methods)
    Organization Software Management Plan, per SLP 1011
    Organization Business Plans (past year, current year)
    Organization Conduct of Operations Documentation (past year, current year)
20. Subcontract Management Plans / Contracts
21. Subcontractor Statement of Work Documents
22. Any documentation from subcontract project reviews

KPA.05 Software Quality Assurance

23. Organization software quality policy
24. Organization software quality definition (goals, quality factors, measures of quality)
25. Software quality assurance plan template
26. Project software quality assurance plan
27. Software qualification evaluation plan
28. Special purpose quality assessment / evaluation plans and results
    Safety, security, vulnerability, reliability
29. Automated support tools for verification / validation of quality characteristics

KPA.06 Software Configuration Management

30. 1. Software configuration management plan template
31. 2. Project software configuration management plan
32. 3. Automated support tools for configuration management

Level 3 Defined

KPA.07 Organization Process Focus

1. O*rganization's software process assets* - A collection of entities, maintained by an organization, for use by projects in developing, tailoring, maintaining, and implementing their software processes. These software process assets typically include:
   the organization's standard software process,
   descriptions of the software life cycles approved for use,
   the guidelines and criteria for tailoring the organization's standard software process,
   the organization's software process database, and
   a library of software process-related documentation.
2. Organization infrastructure
   Software engineering process group charter
   Software quality improvement teams (theme, results, members)
   Examples of personnel participation in teams both internal and external to the organization
   (ES&H, Quality, Software Management Program, SQAS)
3. Previous software assessment results
   Findings report
   Measures of improvement
4. Training plans for organization personnel
   Example plan
   Example courses available (INTEC, SURETY, External), and taken
5. Organizational specific documents
   Organizational policy on Software Management
   Organization Software Management Plan
   Organization Business Plans (past year, current year)
   Organization Conduct of Operations Documentation (past year, current year)

KPA.08 Organization Process Definition

1. O*rganization's software process assets* - A collection of entities, maintained by an organization, for use by projects in developing, tailoring, maintaining, and implementing their software processes. These software process assets typically include:
    the organization's standard software process,
    descriptions of the software life cycles approved for use,
    the guidelines and criteria for tailoring the organization's standard software process,
    the organization's software process database, and
    a library of software process-related documentation.
2. Organization software support environment
    Automated life cycle tool platform
    Network linkage (internal and external)
3. Organizational specific documents
    Organizational policy on Software Management
    Organization Software Management Plan
    Organization Business Plans (past year, current year)
    Organization Conduct of Operations Documentation (past year, current year)

KPA.09 Training Program

1. Organization training plan for personnel
2. Project training plan
3. Personnel training records
4. External training courses taken
5. Internal (INTEC, SURETY) courses taken
6. On-job-training / informal mentorship

KPA.10 Integrated Software Management

7. Organization process definition
8. Software development plan as example of tailoring organization process definition
9. Software reviews documentation

KPA.11 Software Product Engineering

10. Software product engineering process definition
    Part of organization process assets
    Standards & guidelines for the project
11. Project software engineering products
    Software requirements specification
    Software design specification
    Software code examples
    Software test plan
    Software development folder
    Software quality plan
    Software configuration management plan
    Software surety (safety, security, reliability) plans
12. Standards / guidelines (organizational and others)
    policies such as DOE Technical Business Practices
    software guidelines
    software preferred process for software development
    methodology for software development
    Software inspection guidelines and templates
    Software documentation templates
    Software coding standards
13. Software product engineering environment
    Platform architecture description; automated tool support for requirements, design, test, cost estimation

KPA.12 Intergroup Coordination

1. Project intergroup plans, meeting minutes
    Systems engineering, quality engineering / assessment
    Qualification Evaluation Team activities
    rmal review presentations

Hardware / software intergroup requirements analyses
System capability testing, system testing plans and results

2. Specialty engineering coordination
   Security vulnerability, safety, reliability analyses
3. Software work products for coordination with other engineering groups
   Memorandum of understanding, statements of work
   Meeting minutes
   Design analyses / decisions
   Risk analyses

KPA.13 Peer Reviews

1. Software peer review guidelines and templates, including software inspections
2. Project software peer review results
3. Software peer review measures
4. Training on software peer reviews
5. Automated tools support for software peer reviews

Appendix C
General Software Process Improvement Terminology

(Based on CMU/DEI-93-TR-25, Appendix B)

ability to perform - (See *common features*.)

acceptance criteria - The criteria that a system or component must satisfy in order to be accepted by a user, customer, or other authorized entity. [IEEE-STD-610]

acceptance testing - Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. [IEEE-STD-610]

activity - Any step taken or function performed, both mental and physical, toward achieving some objective. Activities include all the work the managers and technical staff do to perform the tasks of the project and organization. (See *task* for contrast.)

activities performed - (See *common features*.)

action item- (1) A unit in a list that has been assigned to an individual or group for disposition. (2) An action proposal that has been accepted.

action proposal- A documented suggestion for change to a process or process-related item that will prevent the future occurrence of defects identified as a result of defect prevention activities. (See also *software process improvement proposal*.)

allocated requirements - (See *system requirements allocated to software*.)

application domain - A bounded set of related systems (i.e., systems that address a particular type of problem). Development and maintenance in an application domain usually requires special skills and/or resources. Examples include payroll and personnel systems, command and control systems, compilers, and expert systems.

assessment - (See *software process assessment*.)

audit - An independent examination of a work product or set of work products to assess compliance with specifications, standards, contractual agreements, or other criteria. [IEEE-STD-610]

baseline - A specification or product that has been formally reviewed and agreed upon, that thereafter serves as the basis for further development, and that can be changed only through formal change control procedures. [IEEE-STD-610]

baseline configuration management- The establishment of baselines that are formally reviewed and agreed on and serve as the basis for further development. Some software work products, e.g., the software design and the code, should have baselines established at predetermined points, and a rigorous change control process should be applied these items. These baselines provide control and stability when interacting with the customer. (See also *baseline management*.)

baseline management- In configuration management, the application of technical and administrative direction to designate the documents and changes to those documents that formally identify and establish baselines at specific times during the life cycle of a configuration item. [IEEE-STD-610]

benchmark - A standard against which measurements or comparisons can be made. [IEEE-STD-610]

bidder - An individual, partnership, corporation, or association that has submitted a proposal and is a candidate to be awarded a contract to design, develop, and/or manufacture one or more products.

capability maturity model - A description of the stages through which software organizations evolve as they define, implement, measure, control, and improve their software processes. This model provides a guide for selecting process improvement strategies by facilitating the determination of current process capabilities and the identification of the issues most critical to software quality and process improvement.

causal analysis - The analysis of defects to determine their underlying root cause.

causal analysis meeting - A meeting, conducted after completing a specific task, to analyze defects uncovered during the performance of that task.

CMM - Acronym for *capability maturity model.*

commitment - A pact that is freely assumed, visible, and expected to be kept by all parties.

commitment to perform - (See *common features*.)

common cause (of a defect) - A cause of a defect that is inherently part of a process or system. Common causes affect every outcome of the process and everyone working in the process. (See *special cause* for contrast.)

common features - The subdivision categories of the CMM key process areas. The common features are attributes that indicate whether the implementation and institutionalization of a key process area is effective, repeatable, and lasting. The CMM common features are the following:

commitment to perform - The actions the organization must take to ensure that the process is established and will endure. Commitment to Perform typically involves establishing organizational policies and senior management sponsorship.

ability to perform - The preconditions that must exist in the project or organization to implement the software process competently. Ability to Perform typically involves resources, organizational structures, and training.

activities performed - A description of the roles and procedures necessary to implement a key process area. Activities Performed typically involve establishing plans and procedures, performing the work, tracking it, and taking corrective actions as necessary.

measurement and analysis - A description of the need to measure the process and analyze the measurements. Measurement and Analysis typically includes examples of the measurements that could be taken to determine the status and effectiveness of the Activities Performed.

verifying implementation - The steps to ensure that the activities are performed in compliance with the process that has been established. Verification typically encompasses reviews and audits by management and software quality assurance.

configuration - In configuration management, the functional and physical characteristics of hardware or software as set forth in technical documentation or achieved in a product. [IEEE-STD-610]

configuration control - An element of configuration management, consisting of the evaluation, coordination, approval or disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification. [IEEE-STD-610]

configuration identification - An element of configuration management, consisting of selecting the configuration items for a system and recording their functional and physical characteristics in technical documentation. [IEEE-STD-610]

configuration item - An aggregation of hardware, software, or both, that is designated for configuration management and treated as a single entity in the configuration management process. [IEEE-STD-610]

configuration management - A discipline applying technical and administrative direction and surveillance to identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements. [IEEE-STD-610]

configuration management library system - The tools and procedures to access the contents of the software baseline library.

configuration unit - The lowest level entity of a configuration item or component that can be placed into, and retrieved from, a configuration management library system.

consistency - The degree of uniformity, standardization, and freedom from contradiction among the documents or parts of system or component. [IEEE-STD-610]

contingency factor - An adjustment (increase) of a size, cost, or schedule plan to account for likely underestimates of these parameters due to incomplete specification, inexperience in estimating the application domain, etc.

contract terms and conditions - The stated legal, financial, and administrative aspects of a contract.

critical computer resource - The parameters of the computing resources deemed to be a source of risk to the project because the potential need for those resources may exceed the amount that is available. Examples include target computer memory and host computer disk space.

critical path - A series of dependent tasks for a project that must be completed as planned to keep the entire project on schedule.

customer - The individual or organization that is responsible for accepting the product and authorizing payment to the developing organization.

defect - A flaw in a system or system component that causes the system or component to fail to perform its required function. A defect, if encountered during execution, may cause a failure of the system.

defect density - The number of defects identified in a product divided by the size of the product component (expressed in standard measurement terms for that product).

defect prevention - The activities involved in identifying defects or potential defects and preventing them from being introduced into a product.

defect root cause - The underlying reason (e.g., process deficiency) that allowed a defect to be introduced.

defined level - (See *maturity level.*)

defined software process - (See *project's defined software process.*)

dependency item - A product, action, piece of information, etc., that must be provided by one individual or group to a second individual or group so that the second individual or group can perform a planned task.

developmental configuration management - The application of technical and administrative direction to designate and control software and associated technical documentation that define the evolving configuration of a software work product during development. Developmental configuration management is under the direct control of the developer. Items under developmental configuration management are not baselines, although they may be baselined and placed under baseline configuration management at some point in their development.

deviation - A noticeable or marked departure from the appropriate norm, plan, standard, procedure, or variable being reviewed.

documented procedure - (See *procedure.*)

effective process - A process that can be characterized as practiced, documented, enforced, trained, measured, and able to improve. (See also *well-defined process.*)

end user - The individual or group who will use the system for its intended operational use when it is deployed in its environment.

end user representatives - A selected sample of end users who represent the total population of end users.

engineering group - A collection of individuals (both managers and technical staff) representing an engineering discipline. Examples of engineering disciplines include systems engineering, hardware engineering, system test, software engineering, software configuration management, and software quality assurance.

evaluation - (See *software capability evaluation.*)

event-driven review/activity - A review or activity that is performed based on the occurrence of an event within the project (e.g., a formal review or the completion of a life cycle stage). (See *periodic review/activity* for contrast.)

findings - The conclusions of an assessment, evaluation, audit, or review that identify the most important issues, problems, or opportunities within the area of investigation.

first-line software manager - A manager who has direct management responsibility (including providing technical direction and administering the personnel and salary functions) for the staffing and activities of a single organizational unit (e.g., a department or project team) of software engineers and other related staff.

formal review - A formal meeting at which a product is presented to the end user, customer, or other interested parties for comment and approval. It can also be a review of the management and technical activities and of the progress of the project.

function - A set of related actions, undertaken by individuals or tools that are specifically assigned or fitted for their roles, to accomplish a set purpose or end.

goals - A summary of the key practices of a key process area that can be used to determine whether an organization or project has effectively implemented the key process area. The goals signify the scope, boundaries, and intent of each key process area.

group - The collection of departments, managers, and individuals who have responsibility for a set of tasks or activities. A group could vary from a single individual assigned part time, to several part-time individuals assigned from different departments, to several individuals dedicated full time.

host computer - A computer used to develop software. (See *target computer* for contrast.)

initial level - (See *maturity level.*)

institutionalization - The building of infrastructure and corporate culture that support methods, practices, and procedures so that they are the ongoing way of doing business, even after those who originally defined them are gone.

integrated software management - The unification and integration of the software engineering and management activities into a coherent defined software process based on the organization's standard software process and related process assets.

integration - (See *software integration*.)

key practices - The infrastructures and activities that contribute most to the effective implementation and institutionalization of a key process area.

key process area - A cluster of related activities that, when performed collectively, achieve a set of goals considered important for establishing process capability. The key process areas have been defined to reside at a single maturity level. They are the areas identified by the SEI to be the principal building blocks to help determine the software process capability of an organization and understand the improvements needed to advance to higher maturity levels. The Level 2 key process areas in the CMM are Requirements Management, Software Project Planning, Software Project Tracking and Oversight, Software Subcontract Management, Software Quality Assurance, and Software Configuration Management. The Level 3 key process areas in the CMM are Organization Process Focus, Organization Process Definition, Training Program, Integrated Software Management, Software Product Engineering, Intergroup Coordination, and Peer Reviews. The Level 4 key process areas are Quantitative Process Management and Software Quality Management. The Level 5 key process areas are Defect Prevention, Technology Change Management, and Process Change Management.

life cycle - (See *software life cycle*.)

maintenance - The process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment. [IEEE-STD-610]

managed and controlled - The process of identifying and defining software work products that are not part of a baseline and, therefore, are not placed under configuration management but that must be controlled for the project to proceed in a disciplined manner. "Managed and controlled" implies that the version of the work product in use at a given time (past or present) is known (i.e., version control), and changes are incorporated in a controlled manner (i.e., change control).

managed level - (See *maturity level.*)

manager - A role that encompasses providing technical and administrative direction and control to individuals performing tasks or activities within the manager's area of responsibility. The traditional functions of a manager include planning, resourcing, organizing, directing, and controlling work within an area of responsibility.

maturity level - A well-defined evolutionary plateau toward achieving a mature software process. The five maturity levels in the SEI's Capability Maturity Model are:

initial - The software process is characterized as ad hoc, and occasionally even chaotic. Few processes are defined, and success depends on individual effort.

repeatable - Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

defined -The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization's standard software process for developing and maintaining software.

managed - Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.

optimizing - Continuous process improvement is enabled by quantitative feedback from the process and from piloting innovative ideas and technologies.

maturity questionnaire - A set of questions about the software process that sample the key practices in each key process area of the CMM. The maturity questionnaire is used as a springboard to appraise the capability of an organization or project to execute a software process reliably.

measure - A unit of measurement (such as source lines of code or document pages of design).

measurement - The dimension, capacity, quantity, or amount of something (e.g., 300 source lines of code or 7 document pages of design).

method - A reasonably complete set of rules and criteria that establish a precise and repeatable way of performing a task and arriving at a desired result.

methodology - A collection of methods, procedures, and standards that defines an integrated synthesis of engineering approaches to the development of a product.

milestone - A scheduled event for which some individual is accountable and that is used to measure progress.

nontechnical requirements - Agreements, conditions and / or contractual terms that affect and determine the management activities of a software project.

operational software - The software that is intended to be used and operated in a system when it is delivered to its customer and deployed in its intended environment.

optimizing level - (See *maturity level.*)

organization - A unit within a company or other entity (e.g., government agency or branch of service) within which many projects are managed as a whole. All projects within an organization share a common top-level manager and common policies.

organization's measurement program - The set of related elements for addressing an organization's measurement needs. It includes the definition of organization-wide measurements, methods and practices for collecting organizational measurement data, methods and practices for analyzing organizational measurement data, and measurement goals for the organization.

organization's software process assets - A collection of entities, maintained by an organization, for use by projects in developing, tailoring, maintaining, and implementing their software processes. These software process assets typically include:

the organization's standard software process,
descriptions of the software life cycles approved for use,
the guidelines and criteria for tailoring the organization's standard software process,
the organization's software process database, and
a library of software process-related documentation.

Any entity that the organization considers useful in performing the activities of process definition and maintenance could be included as a process asset.

organization's software process database - A database established to collect and make available data on the software processes and resulting software work products, particularly as they relate to the organization's standard software process. The database contains or references both the actual measurement data and the related information needed to understand the measurement data and assess it for reasonableness and applicability. Examples of process and work product data include estimates of software size, effort, and cost; actual data on software size, effort, and cost; productivity data; peer review coverage and efficiency; and number and severity of defects found in the software code.

organization's standard software process - The operational definition of the basic process that guides the establishment of a common software process across the software projects in an organization. It describes the fundamental software process elements that each software project is expected to incorporate into its defined software process. It also describes the relationships (e.g., ordering and interfaces) between these software process elements.

orientation - An overview or introduction to a topic for those overseeing or interfacing with the individuals responsible for performing in the topic area. (See *train* for contrast.)

Pareto analysis - The analysis of defects by ranking causes from most significant to least significant. Pareto analysis is based on the principle, named after the 19th-century economist Vilfredo Pareto, that most effects come from relatively few causes, i.e., 80% of the effects come from 20% of the possible causes.

peer review - A review of a software work product, following defined procedures, by peers of the producers of the product for the purpose of identifying defects and improvements.

peer review leader - An individual specifically trained and qualified to plan, organize, and lead a peer review.

periodic review/activity - A review or activity that occurs at specified regular time intervals. (See *event-driven review/activity* for contrast.)

policy - A guiding principle, typically established by senior management, which is adopted by an organization or project to influence and determine decisions.

prime contractor - An individual, partnership, corporation, or association that administers a subcontract to design, develop, and / or manufacture one or more products.

procedure - A written description of a course of action to be taken to perform a given task. [IEEE-STD-610]

process - A sequence of steps performed for a given purpose; for example, the software development process. [IEEE-STD-610]

process capability - The range of expected results that can be achieved by following a process. (See *process performance* for contrast.)

process capability baseline - A documented characterization of the range of expected results that would normally be achieved by following a specific process under typical circumstances. A process capability baseline is typically established at an organizational level. (See *process performance baseline* for contrast.)

process database - (See *organization's software process database*.)

process description- The operational definition of the major components of a process. Documentation that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a process. It may also include the procedures for determining whether these provisions have been satisfied. Process descriptions may be found at the task, project, or organizational level.

process development- The act of defining and describing a process. It may include planning, architecture, design, implementation, and validation.

process measurement - The set of definitions, methods, and activities used to take measurements of a process and its resulting products for the purpose of characterizing and understanding the process.

process performance - A measure of the actual results achieved by following a process. (See *process capability* for contrast.)

process performance baseline - A documented characterization of the actual results achieved by following a process, which is used as a benchmark for comparing actual process performance against expected process performance. A process performance baseline is typically established at the project level, although the initial process performance baseline will usually be derived from the process capability baseline. (See *process capability baseline* for contrast.)

process tailoring - The activity of creating a process description by elaborating, adapting, and/or completing the details of process elements or other incomplete specifications of a process. Specific business needs for a project will usually be addressed during process tailoring.

product - (See *software product* and *software work product*.)

profile - A comparison, usually in graphical form, of plans or projections versus actuals, typically over time.

project - An undertaking requiring concerted effort, which is focused on developing and/or maintaining a specific product. The product may include hardware, software, and other components. Typically a project has its own funding, cost accounting, and delivery schedule.

project's defined software process - The operational definition of the software process used by a project. The project's defined software process is a well-characterized and understood software process, described in terms of software standards, procedures, tools, and methods. It is developed by tailoring the organization's standard software process to fit the specific characteristics of the project. (See also *organization's standard software process, effective process,* and *well-defined process*.)

project manager - The role with total business responsibility for an entire project; the individual who directs, controls, administers, and regulates a project building a software or hardware/software system. The project manager is the individual ultimately responsible to the customer.

project software manager - The role with total responsibility for all the software activities for a project. The project software manager is the individual the project manager deals with in terms of software commitments and who controls all the software resources for a project.

quality - (1) The degree to which a system, component, or process meets specified requirements. (2) The degree to which a system, component, or process meets customer or user needs or expectations. [IEEE-STD-610]

quality assurance - (See *software quality assurance*.)

quantitative control - Any quantitative or statistically-based technique appropriate to analyze a software process, identify special causes of variations in the performance of the software process, and bring the performance of the software process within well-defined limits.

repeatable level - (See *maturity level.*)

required training - Training designated by an organization to be required to perform a specific role.

risk - Possibility of suffering loss.

risk management - An approach to problem analysis which weighs risk in a situation by using risk probabilities to give a more accurate understanding of the risks involved. Risk management includes risk identification, analysis, prioritization, and control.

risk management plan - The collection of plans that describe the risk management activities to be performed on a project.

role - A unit of defined responsibilities that may be assumed by one or more individuals.

SCE - Acronym for *software capability evaluation*.

SCM - Acronym for *software configuration management*.

senior manager - A management role at a high enough level in an organization that the primary focus is the long-term vitality of the organization, rather than short-term project and contractual concerns and pressures. In general, a senior manager for engineering would have responsibility for multiple projects.

software architecture - The organizational structure of the software or module. [IEEE-STD-610]

software baseline audit - An examination of the structure, contents, and facilities of the software baseline library to verify that baselines conform to the documentation that describes the baselines.

software baseline library - The contents of a repository for storing configuration items and the associated records.

software build - An operational version of a software system or component that incorporates a specified subset of the capabilities the final software system or component will provide. [IEEE-STD-610]

software capability evaluation - An appraisal by a trained team of professionals to identify contractors who are qualified to perform the software work or to monitor the state of the software process used on an existing software effort.

software configuration control board - A group responsible for evaluating and approving or disapproving proposed changes to configuration items, and for ensuring implementation of approved changes.

software development plan - The collection of plans that describe the activities to be performed for the software project. It governs the management of the activities performed by the software engineering group for a software project. It is not limited to the scope of any particular planning standard, such as DOD-STD-2167A and IEEE-STD-1058, which may use similar terminology.

software engineering group - The collection of individuals (both managers and technical staff) who have responsibility for software development and maintenance activities (i.e., requirements analysis, design, code, and test) for a project. Groups performing software-related work, such as the software quality assurance group, the software configuration management group, and the software engineering process group, are not included in the software engineering group.

software engineering process group - A group of specialists who facilitate the definition, maintenance, and improvement of the software process used by the organization. In the key practices, this group is generically referred to as "the group responsible for the organization's software process activities."

software engineering staff - The software technical people (e.g., analysts, programmers, and engineers), including software task leaders, who perform the software development and maintenance activities for the project, but who are not managers.

software integration - A process of putting together selected software components to provide the set or specified subset of the capabilities the final software system will provide.

software life cycle - The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and, sometimes, retirement phase. [IEEE-STD-610]

software manager - Any manager, at a project or organizational level, who has direct responsibility for software development and/or maintenance.

software plans - The collection of plans, both formal and informal, used to express how software development and/or maintenance activities will be performed. Examples of plans that could be included: software development plan, software quality assurance plan, software configuration management plan, software test plan, risk management plan, and process improvement plan.

software process - A set of activities, methods, practices, and transformations that people use to develop and maintain software and the associated products (e.g., project plans, design documents, code, test cases, and user manuals).

software process assessment - An appraisal by a trained team of software professionals to determine the state of an organization's current software process, to determine the high-priority software process-related issues facing an organization, and to obtain the organizational support for software process improvement.

software process assets - (See *organization's software process assets.*)

software process capability - (See *process capability.*)

software process description - The operational definition of a major software process component identified in the project's defined software process or the organization's standard software process. It documents, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a software process. (See also *process description*.)

software process element - A constituent element of a software process description. Each process element covers a well-defined, bounded, closely related set of tasks (e.g., software estimating element, software design element, coding element, and peer review element). The descriptions of the process elements may be templates to be filled in, fragments to be completed, abstractions to be refined, or complete descriptions to be modified or used unmodified.

software process improvement plan - A plan, derived from the recommendations of a software process assessment, that identifies the specific actions that will be taken to improve the software process and outlines the plans for implementing those actions. Sometimes referred to as an action plan.

software process improvement proposal - A documented suggestion for change to a process or process-related item that will improve software process capability and performance. (See also *action proposal.*)

software process maturity - The extent to which a specific process is explicitly defined, managed, measured, controlled, and effective. Maturity implies a potential for growth in capability and indicates both the richness of an organization's software process and the consistency with which it is applied in projects throughout the organization.

software process performance - (See *process performance.*)

software process-related documentation - Example documents and document fragments, which are expected to be of use to future projects when they are tailoring the organization's standard software process. The examples may cover subjects such as a project's defined software process, standards, procedures, software development plans, measurement plans, and process training materials.

software product - The complete set, or any of the individual items of the set, of computer programs, procedures, and associated documentation and data designated for delivery to a customer or end user. [IEEE-STD-610] (See *software work product* for contrast.)

software project - An undertaking requiring concerted effort, which is focused on analyzing, specifying, designing, developing, testing, and/or maintaining the software components and associated documentation of a system. A software project may be part of a project building a hardware/software system.

software quality assurance - (1) A planned and systematic pattern of all actions necessary to provide adequate confidence that a software work product conforms to established technical requirements. (2) A set of activities designed to evaluate the process by which software work products are developed and/or maintained. [Derived from IEEE-STD-610]

software quality goal - Quantitative quality objectives defined for a software work product.

software quality management - The process of defining quality goals for a software product, establishing plans to achieve these goals, and monitoring and adjusting the software plans, software work products, activities, and quality goals to satisfy the needs and desires of the customer and end users.

software-related group - A collection of individuals (both managers and technical staff) representing a software engineering discipline that supports, but is not directly responsible for, software development and/or maintenance. Examples of software engineering disciplines include software quality assurance and software configuration management.

software requirement - A condition or capability that must be met by software needed by a user to solve a problem or achieve an objective. [IEEE-STD-610]

software work product - Any artifact created as part of defining, maintaining, or using a software process, including process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to a customer or end user. (See *software product* for contrast.)

SPA - Acronym for *software process assessment*.

special cause (of a defect) - A cause of a defect that is specific to some transient circumstance and not an inherent part of a process. Special causes provide random variation (noise) in process performance. (See *common cause* for contrast.)

SQA - Acronym for *software quality assurance.*

staff - The individuals, including task leaders, who are responsible for accomplishing an assigned function, such as software development or software configuration management, but who are not managers.

stage - A partition of the software effort that is of a manageable size and that represents a meaningful and measurable set of related tasks which are performed by the project. A stage is usually considered a subdivision of a software life cycle and is often ended with a formal review prior to the onset of the following stage.

standard - Mandatory requirements employed and enforced to prescribe a disciplined uniform approach to software development. [Derived from IEEE-STD-610]

standard software process - (See *organization's standard software process*.)

statement of work - A description of all the work required to complete a project, which is provided by the customer.

subcontract manager - A manager in the prime contractor's organization who has direct responsibility for administering and managing one or more subcontracts.

subcontractor - An individual, partnership, corporation, or association that contracts with an organization (i.e., the prime contractor) to design, develop, and/or manufacture one or more products.

system - A collection of components organized to accomplish a specific function or set of functions. [IEEE-STD-610]

system engineering group - The collection of individuals (both managers and technical staff) who have responsibility for specifying the system requirements; allocating the system requirements to the hardware, software, and other components; specifying the interfaces between the hardware, software, and other components; and monitoring the design and development of these components to ensure conformance with their specifications.

system requirement - A condition or capability that must be met or possessed by a system or system component to satisfy a condition or capability needed by a user to solve a problem. [IEEE-STD-610]

system requirements allocated to software - The subset of the system requirements that are to be implemented in the software components of the system. The allocated requirements are a primary input to the software development plan. Software requirements analysis elaborates and refines the allocated requirements and results in software requirements which are documented.

tailor - To modify a process, standard, or procedure to better match process or product requirements.

target computer - The computer on which delivered software is intended to operate. (See *host computer* for contrast.)

task - (1) A sequence of instructions treated as a basic unit of work. [IEEE-STD-610] (2) A well-defined unit of work in the software process that provides management with a visible checkpoint into the status of the project. Tasks have readiness criteria (preconditions) and completion criteria (postconditions). (See *activity* for contrast.)

task kick-off meeting - A meeting held at the beginning of a task of a project for the purpose of preparing the individuals involved to perform the activities of that task effectively.

task leader - The leader of a technical team for a specific task, who has technical responsibility and provides technical direction to the staff working on the task.

team - A collection of people, often drawn from diverse but related groups, assigned to perform a well-defined function for an organization or a project. Team members may be part-time participants of the team and have other primary responsibilities.

testability - (1) The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. (2) The degree to which a requirement is stated in terms that permit establishment of test criteria and performance of tests to determine whether those criteria have been met. [IEEE-STD-610] *technical requirements* - Those requirements that describe what the software must do and its operational constraints. Examples of technical requirements include functional, performance, interface, and quality requirements.

technology - The application of science and / or engineering in accomplishing some particular result.

traceability - The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another. [IEEE-STD-610]

train - To make proficient with specialized instruction and practice. (See also *orientation*.)

training group - The collection of individuals (both managers and staff) who are responsible for coordinating and arranging the training activities for an organization. This group typically prepares and conducts most of the training courses and coordinates use of other training vehicles.

training program - The set of related elements that focus on addressing an organization's training needs. It includes an organization's training plan, training materials, development of training, conduct of training, training facilities, evaluation of training, and maintenance of training records.

training waiver - A written approval exempting an individual from training that has been designated as required for a specific role. The exemption is granted because it has been objectively determined that the individual already possesses the needed skills to perform the role.

unit - (1) A separately testable element specified in the design of a computer software component. (2) A logically separable part of a computer program. (3) A software component that is not subdivided into other components. [IEEE-STD-610]

user- (See *end user.)*

validation- The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements. [IEEE-STD-610]

verification- The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. [IEEE-STD-610]

verifying implementation - (See *common features*.)

waiver - (See *training waiver)*.

well-defined process - A process that includes readiness criteria, inputs, standards and procedures for performing the work, verification mechanisms (such as peer reviews), outputs, and completion criteria. (See also *effective process*.)